Minutes and Expansions of
November 8-9, 1999, Interagency Software Meeting at NCAR.

Attendees from NCAR, NASA/DAO, LLNL, LANL, and ORNL.

Richard B. Rood, NASA/Data Assimilation Office, (Appointed lead of first phase of the groups activities.)

In this meeting we had initial discussions about the development of the process and infrastructure to allow interested scientists from each of the above organizations to work more effectively on joint development of climate models. In particular, since each of the non-NCAR organizations and subsets within NCAR are working with the Climate System Model, how can these collaborations be more substantive and more productive? All of the parties agreed that a more formal approach to software development was needed, including more actively addressing the issues of useability and performance on distributed-memory, distributed-processor computers. It was also recognized that the path forward was difficult, confronting traditional individuality of the scientific community. Nevertheless, there is an underlying feeling by many in the group that problems of institutional collaboration must be faced if the U.S. climate modeling community is to maintain a competitive stature with efforts from other countries.

Broadly, the members from the different institutions agreed to pursue the following goals.

- Develop the capability to allow concurrent development in a controlled environment, including the facilitation of controlled experimentation by a diverse user community.

- Identification of a clear path of migration from discovery-driven research activities, including scientists at dispersed universities, to the more mission or product-oriented developments at the national labs, and in particular at NCAR. Success in this activity will require identification of testing and validation processes, as well as consideration of computational performance.

- Improve the integration of software, systems (hardware), and science requirements to assure more robust software and computational platforms optimized with respect to both scientific quality and completeness. Success in this activity addresses issues of maintainability, portability, evolution and performance.

In addition, the attending parties agreed to work together to develop the decision making process that is needed to support the development of complex computer codes for multiple applications by distributed partners. Issues of collaborative design and requirements definition must also be faced. The first task of the group was to identify the elements of a software infrastructure that would allow the above goals to be addressed.

While each of the interested organizations has as an ultimate goal addressing various issues of Earth-science, the tangible commodity produced by each organization is software. This software represents the scientific elements of earth system as well as the

ancillary software needed for diagnostics and quality control and setup and management of input and output data sets. The system is intrinsically complex, requiring on the order of a half a million lines of code containing many hundreds, perhaps thousands, of logical elements. There are diverse approaches to the implementation of every functionality in the complete system. Therefore, if the above goals are to be met, more formal management and control of the software is needed.

The type and level of control is not uniform across the system. Some functions are relatively mature with little controversy and change on only long time scales. These functions are reasonable targets for standardization and sharing across all institutions, with the hope of reducing the total resources that are spent in maintaining these functions. Other functions are less well defined and the subject of discovery-research activities with concomitant levels of volatility. These functions are intrinsically unmanageable, but with more formal definitions of interfaces and tests, we feel the development environment can facilitate controlled experimentation.

In addition, because of profound changes in the computing environment, the interactions of the applications software with the computing environment are becoming more complex and fragile. This includes changes in computational architectures, as well as in support software provided by vendors. If the software to accomplish earth simulation and assimilation is to remain viable, then these interactions with the computational system need to be integrated more thoroughly in the code development process. In short, the intellectual contributions of software experts need to be brought to the same decision making level as the scientific decisions. Science capabilities must be optimized in the consideration of these software issues; however, the software issues must not be deemed subsidiary.

Within the discussions of the meeting, the following questions were raised as exemplary or a subset of the issues that must be addressed in the first phase of the committee's activities.

Does successful concurrent development require joint ownership and management of a single repository? Are there other strategies of distributed repositories with regular merges?

> There is no doubt that the success of concurrent development requires a commitment to develop rules of management and process. We recognize the need for more formalized process and that the tenets of software engineering must be appropriately tuned to the development of scientific codes.
>
> Issues that must be addressed:
>
> What is missing in the current process?
> Role of testing.
> Documentation.
> Distributed versus centralized management.

What is the build process.
How is feedback from experiments communicated back to core team and all interested organizations?

What level of configuration management is needed?

The need for formal design

Science
Computational
I/O
Diagnostics

What is the role of quality assessment

Code
Product
Validation

What are appropriate standards?

Porting standards
Facilitate exchange
Mitigate risk
    Interaction with middleware
    Simplicity versus sophistication
Roundoff versus zero diffs

Can we develop the function of a Professional Society that endorses standards and then have agencies reward the adherence to standards in their funding decisions?

How do we develop a forward looking function that mitigates risk of changing hardware and software industry.

Where can we find a pool of software experts to help in the design of the infrastructure?

What are appropriate prototype efforts?